# Game State and Evaluator Report Architecture

## Design

### Architecture

Important Component in Design and Interface provided by that classes

Game

Provide logic function

- getAlly
- getOpponent
- getNearby

Mannage Data in Hex grid

- moveMinionByDirection (Minion, Direction)
- buyHexAt(Leader buyer, (int, int))
- getHexAt((int, int))
- getHexOwner((int, int))
- attackTo(Minion attacker, Direction direction, long damage)
- spawnMinionAt((int, int), String type, Leader owner)

Minion

Class represent minion

- move(Direction)
- attack(Direction, long damage)
- getDamage(long damage)
- getHealth()
- getDefense()
- getOwner()

- getGame()
- getMinionType()

Hex

- setOwner()
- getOwner()
- getAttack()
- removeMinionOnHex()
- hasMinionOnHex()
- getMinionOnHex()

Leader

Provide method that user to call when game change state

- turnBegin()
- turnEnd()
- spawnMinionState()
- buyHexState()

Some action method that use to interact with game

- reduceBudget()
- executeMinionsStrategy()
- buyHex()
- spawnMinionAt()

and Getter, Setter

Strategy

- execute(Minion)

StrategyParser

- parse()

StrategyTokenizer

- comsume()
- peek()

Direction

Enum that use to tell direction on hex grid

Provide some utility function

- transformDirection()

## Thing that we learned from design

- Parser and Evaluater is very powerful to interpred langauge of gramma, it can give program to do something more flexible
- Desing without design pattern can be so hard
- and not relate to design wrtiting doc, report is so deadly hard